SYSTEM AND METHOD FOR MEMORY HUB-BASED EXPANSION BUS

TECHNICAL FIELD

[001] The present invention relates generally to a memory system for a processor-based computing system, and more particularly, to a hub-based memory system providing expansion capabilities for computer components.

BACKGROUND OF THE INVENTION

[002] Computer systems use memory devices, such as dynamic random access memory ("DRAM") devices, to store data that are accessed by a processor. These memory devices are normally used as system memory in a computer system. In a typical computer system, the processor communicates with the system memory through a processor bus and a memory controller. The memory devices of the system memory, typically arranged in memory modules having multiple memory devices, are coupled through a memory bus to the memory controller. The processor issues a memory request, which includes a memory command, such as a read command, and an address designating the location from which data or instructions are to be read. The memory controller uses the command and address to generate appropriate command signals as well as row and column addresses, which are applied to the system memory through the memory bus. In response to the commands and addresses, data are transferred between the system memory and the processor. The memory controller is often part of a system controller, which also includes bus bridge circuitry for coupling the processor bus to an expansion bus, such as a PCI bus.

[003] In memory systems, high data bandwidth is desirable. Generally, bandwidth limitations are not related to the memory controllers since the memory controllers sequence data to and from the system memory as fast as the memory devices allow. One approach that has been taken to increase bandwidth is to increase the speed of the memory data bus coupling the memory controller to the memory devices. Thus, the same amount of information can be moved over the memory data bus in less time. However, despite increasing memory data bus speeds, a corresponding increase in bandwidth does not result.

One reason for the non-linear relationship between data bus speed and bandwidth is the hardware limitations within the memory devices themselves. That is, the memory controller has to schedule all memory commands to the memory devices such that the hardware limitations are honored. Although these hardware limitations can be reduced to some degree through the design of the memory device, a compromise must be made because reducing the hardware limitations typically adds cost, power, and/or size to the memory devices, all of which are undesirable alternatives. Thus, given these constraints, although it is easy for memory devices to move "well-behaved" traffic at ever increasing rates, for example, sequel traffic to the same page of a memory device, it is much more difficult for the memory devices to resolve "badly-behaved traffic," such as bouncing between different pages or banks of the memory device. As a result, the increase in memory data bus bandwidth does not yield a corresponding increase in information bandwidth.

[004] In addition to the limited bandwidth between processors and memory devices, the performance of computer systems is also limited by latency problems that increase the time required to read data from system memory devices. More specifically, when a memory device read command is coupled to a system memory device, such as a synchronous DRAM ("SDRAM") device, the read data are output from the SDRAM device only after a delay of several clock periods. Therefore, although SDRAM devices can synchronously output burst data at a high data rate, the delay in initially providing the data can significantly slow the operating speed of a computer system using such SDRAM devices. Increasing the memory data bus speed can be used to help alleviate the latency issue. However, as with bandwidth, the increase in memory data bus speeds do not yield a linear reduction of latency, for essentially the same reasons previously discussed.

[005] Although increasing memory data bus speed has, to some degree, been successful in increasing bandwidth and reducing latency, other issues are raised by this approach. For example, as the speed of the memory data bus increases, loading on the memory bus needs to be decreased in order to maintain signal integrity since traditionally, there has only been wire between the memory controller and the memory slots into which

the memory modules are plugged. Several approaches have been taken to address the memory bus loading issue. For example, reducing the number of memory slots to limit the number of memory modules that contribute to the loading of the memory bus, adding buffer circuits on a memory module in order to provide sufficient fanout of control signals to the memory devices on the memory module, and providing multiple memory device interfaces on the memory module since there are too few memory module connectors on a single memory device interface. The effectiveness of these conventional approaches are, however, limited. A reason why these techniques were used in the past is that it was cost-effective to do so. However, when only one memory module can be plugged in per interface, it becomes too costly to add a separate memory interface for each memory slot. In other words, it pushes the system controllers package out of the commodity range and into the boutique range, thereby, greatly adding cost.

[006] One recent approach that allows for increased memory data bus speed in a cost effective manner is the use of multiple memory devices coupled to the processor through a memory hub. A computer system 100 shown in Figure 1 uses a memory hub architecture. The computer system 100 includes a processor 104 for performing various computing functions, such as executing specific software to perform specific calculations or tasks. The processor 104 includes a processor bus 106 that normally includes an address bus, a control bus, and a data bus. The processor bus 106 is typically coupled to cache memory 108, which, is typically static random access memory ("SRAM"). Finally, the processor bus 106 is coupled to a system controller 110, which is also sometimes referred to as a bus bridge. The system controller 110 serves as a communications path to the processor 104 for a variety of other components. For example, as shown in Figure 1, the system controller 110 includes a graphics port that is typically coupled to a graphics controller 112, which is, in turn, coupled to a video terminal 114. The system controller 110 is also coupled to one or more input devices 118, such as a keyboard or a mouse, to allow an operator to interface with the computer system 100. Typically, the computer system 100 also includes one or more output devices 120, such as a printer, coupled to the processor 104 through the system controller 110. One or more data storage devices 124 are also typically coupled to the

processor 104 through the system controller 110 to allow the processor 104 to store data or retrieve data from internal or external storage media (not shown). Examples of typical storage devices 124 include hard and floppy disks, tape cassettes, and compact disk read-only memories (CD-ROMs).

[007] The system controller 110 includes a memory hub controller 128 that is coupled to the processor 104. The system controller 110 is further coupled over a high speed bi-directional or unidirectional system controller/hub interface 134 to several memory modules 130a-n. As shown in Figure 1, the controller/hub interface 134 includes a downstream bus 154 and an upstream bus 156 which are used to couple data, address, and/or control signals away from or toward, respectively, the memory hub controller 128. Typically, the memory modules 130a-n are coupled in a point-to-point or daisy chain architecture such that the memory modules 130a-n are connected one to another in series. Thus, the system controller 110 is coupled to a first memory module 130a, with the first memory module 130a connected to a second memory module 130b, and the second memory module 130b coupled to a third memory module 130c, and so on in a daisy chain fashion. Each memory module 130a-n includes a memory hub 140 that is coupled to the system controller/hub interface 134, and is further coupled a number of memory devices 148 through command, address and data buses, collectively shown as local memory bus 150. The memory hub 140 efficiently routes memory requests and responses between the memory hub controller 128 and the memory devices 148.

[008] The memory devices 148 on the memory modules 130a-n are typically capable of operating at high clock frequencies in order to facilitate the relatively high speed operation of the overall memory system. Consequently, computer systems employing this architecture can also use the high-speed system controller/hub interface 134 to complement the high clock speeds of the memory devices 148. Additionally, with a memory hub based system, signal integrity can be maintained on the system controller/hub interface 134 since the signals are typically transmitted through multiple memory hubs 140 to and from the memory hub controller 128. Moreover, this architecture also provides for easy expansion

of the system memory without concern for degradation in signal quality as more memory modules are added, such as occurs in conventional memory bus architectures.

[009] Although the memory hub architecture shown in Figure 1 provides improved memory system performance, the advantages my not directly benefit the various components of the computer system 100. As previously described, the components, such as the graphics controller 112, the input and output devices 118, 120, and the data storage 124 are coupled to the system controller 110. It is through the system controller 110 that the components 112, 118, 120, 124 access the memory modules 130a-n. As a result of the memory requests necessarily being coupled through the system controller 110, a "bottleneck" can often result since the system controller 110 can handle only a finite number of memory requests, and corresponding memory responses from the memory modules 130a-n, at a given time. The graphics port through which the graphics controller 112 is coupled to the system controller 110 provides some relief to the bottleneck issue, since the graphics port typically provides direct memory access (DMA) to the memory modules 130a-n, as well known in the art. That is, the graphics controller 112 is able to access the memory modules 130a-n directly, with limited intervention by the system controller 110.

[010] As well known, arbitration schemes are implemented by the system controller 110 in order to prioritize memory requests it receives from the various components 112, 118, 120, 124, as well as memory requests received from the processor 104. The arbitration schemes that are implemented attempt to provide efficient memory access to the various components 112, 118, 120, 124, and processor 104 in order to maximize processing capabilities. Some memory requests are given priority over others regardless of the order in which the requests are received by the system controller 110, for example, the processor 104 is often given highest priority to access the memory modules 130a-n to avoid the situation where processing is halted while the processor 104 is waiting for a memory request to be serviced. As sophisticated as arbitration techniques have become, it is still unlikely that bottlenecks at the system controller 110 can be completely avoided. Even where a component is given direct memory access to the memory modules 130a-n, such as

the graphics controller 112, it is nevertheless subject to the arbitration routine that is implemented by the system controller 110, and consequently, the component does not have unlimited access privileges to the memory modules 130a-n. It is by the nature of the architecture used in the computer system 100, namely, providing access to the memory modules 130a-n through the single point of the system controller 110, that makes bottlenecks at the system controller 110 inevitable. Therefore, there is a need for an alternative system and method for providing components of a processing system, such as a computer system, access to memory resources.

SUMMARY OF THE INVENTION

[011] A system memory in one aspect of the invention includes a memory hub controller, a memory module accessible by the memory hub controller, and an expansion module coupled to the memory module having a processor circuit also having access to the memory module. The memory hub controller provides memory requests to access memory devices, and the memory module includes a plurality of memory devices coupled to a memory hub. The memory hub receives the memory requests, accesses the memory devices according to the memory requests, and provides memory responses in response to the memory requests. The processor circuit of the expansion module provides memory requests to the memory hub of the memory module to access the memory devices, and processes data returned in the memory responses from the memory hub. The memory hub controller is coupled to the memory hub through a first portion of a memory bus on which the memory requests and the memory responses are coupled. A second portion of the memory bus couples the memory hub to the processor circuit and is used to couple memory requests from the processor circuit and memory responses provided by the memory hub to the processor circuit.

BRIEF DESCRIPTION OF THE DRAWINGS

[012] Figure 1 is a partial block diagram of a conventional processor-based computing system having a memory hub-based system memory.

[013] Figure 2 is a partial block diagram of a processor-based computing system having a memory hub-based memory system according to an embodiment of the present invention providing peripheral component expansion capabilities.

[014] Figure 3 is a partial block diagram of a memory hub of the hub-based memory system of Figure 2.

[015] Figure 4 is a partial block diagram of a processor-based computing system having a memory hub-based memory system according to an alternative embodiment of the present invention providing peripheral component expansion capabilities.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[016] Figure 2 illustrates a processor based computing system 200 according to an embodiment of the present invention. The system 200 includes many of the same functional blocks as previously described with reference to Figure 1. As such, the same reference numbers will be used in Figure 2 as in Figure 1 to refer to the same functional blocks where appropriate. The system 200 includes a processor 104 coupled to a system controller 110 through a processor bus 106. As in Figure 1, the processor performs various computing functions, for example, executing software to perform specific calculations or tasks, and the processor bus 106 typically includes an address bus, a control bus, and a data bus. A cache memory 108 is also coupled to the processor bus 106 to provide the processor 104 with temporary storage of frequently used data and instructions. As previously discussed with respect to Figure 1, the system controller 110 serves as a communications path to the processor 104 for a variety of other components. Typically, this includes one or more input devices 118, such as a keyboard or a mouse, to allow an operator to interface with the system 200, one or more output devices 120, such as a printer, and one or more data storage devices 124 to allow the processor 104 to store data or retrieve data from internal or external storage media (not shown).

[017] As shown in Figure 2, the system controller 110 includes a memory hub controller 128 to which several memory modules 130a-c are coupled over a high speed bi-directional or unidirectional system controller/hub interface 134. The controller/hub

interface 134 includes a downstream bus 154 and an upstream bus 156 which are used to couple data, address, and/or control signals away from or toward, respectively, the memory hub controller 128. As shown in Figure 2, the memory modules 130a-c are coupled in a point-to-point architecture such that the memory modules 130a-c are connected one to another in series. Each memory module 130a-c in the system 200 includes a memory hub 240 that is coupled to the system controller/hub interface 134, and is further coupled a number of memory devices 148 through command, address and data buses, collectively shown as bus 150. As previously mentioned, the memory hub 240 efficiently routes and arbitrates memory requests and responses between the memory hub controller 128 and the memory devices 148. As will be explained in further detail below, the memory hub 240 can receive memory requests and provide memory responses in both downstream and upstream directions over the downstream and upstream buses 154, 156, respectively.

[018] In contrast to the computer system 100 of Figure 1, the system 200 includes a component expansion module 230 coupled to the controller/hub interface 134. As shown in Figure 2, the component expansion module 230 includes a graphics controller 234 coupled to local memory devices 248 over a local graphics/memory bus 250. The graphics controller 234, the local graphics/memory bus 250, and the local memory devices 248 can be of conventional design and operation, as well known in the art. The graphics/memory bus 250 includes command, data, and address buses as well known in the art. A video bus 260 can be used for coupling video data from the graphics controller 234 to a video terminal (not shown) as known in the art. It will be appreciated that the component expansion module 230 replaces the graphics controller 112 of the computer system 100. That is, the component expansion module 230 can provide the computer graphics capabilities and functionality of the graphics controller 112.

[019] Although the component expansion module 230 is shown in Figure 2 as having local memory devices 248, access to data stored in the system memory, such as memory modules 130a-c, is often required for processing by the graphics controller 234. For example, the memory provided by the local memory devices 248 may not be sufficient to store all of the graphics data necessary for rendering a scene. As a result, the bulk of the

graphics data is typically loaded into system memory, with the graphics controller 234 retrieving the portion of graphics data necessary for rendering the current scene from the system memory. Additionally, since access to the local memory devices 248 is typically limited to the graphics controller 234, data that has been first processed elsewhere, for example, by the processor 104, must be stored to a location in the system memory for retrieval by the graphics controller 234 before being stored in the local memory devices 248 for further processing. Thus, access to the memory modules 130a-c by the component expansion module 230 is often necessary.

[020] The arrangement of the system 200 allows for access to the memory modules 130a-c by the component expansion module 230 without intervention by the system controller 110. As previously discussed, the memory hubs 240 can receive memory requests and provide memory responses in both the downstream and upstream directions. By adopting a consistent communication protocol with the memory hubs 240 of the memory modules 130a-c, communication with the memory hubs 240 of the memory modules 130a-c can be performed directly by the component expansion module 230, thereby eliminating the need for intervention by the system controller 110. As a result, access to the memory modules 130a-c is not limited to going through the system controller 110, but the component expansion module 230 can access the memory modules 130a-c directly. In contrast, the graphics controller 112 in the computer system 100 (Figure 1) is typically coupled to the system controller 110 through an advanced graphics port, and although the graphics controller 112 has DMA access to the memory, it is still nevertheless subject to the memory request and memory response loading issues of the system controller 110. In the system 200, however, the graphics controller 234 is not subject to the loading issues of the system controller 110.

[021] Many suitable communication protocols are known in the art, including the use of command packets that include appropriate information for making memory requests to particular memory modules 130a-c in the system 200 and providing memory responses in return. For example, command packets can include information such as identification data for uniquely identifying the particular memory request, address information for identifying

a particular memory module 130a-c to which the memory request is directed, and memory device command information, including memory addresses, command type, and where a write operation is requested, data can be included as well. Other protocols can be used as well, and it will be appreciated by those ordinarily skilled in the art that the present invention is not limited by the particular protocol implemented.

[022] Additionally, the arrangement of the system 200 reduces the memory request and response load on the system controller 110 since it is relieved from handling the memory requests from a requesting entity, namely the graphics controller 112 (Figure 1). For these reasons, the likelihood that a memory request and response bottleneck occurring at the system controller 110 is also reduced. Moreover, by coupling the component expansion module 230 to the controller/hub interface 134 rather than to the system controller 110, the number of buses in the system 200 can be reduced.

[023] Figure 3 illustrates a portion of the memory hub 240 (Figure 2). The memory hub 240 includes four link interfaces 302, 304, 306, 308 coupled to a cross bar switch 310 by respective local link buses 312, 314, 316, 318. Memory controllers 324a, 324b are further coupled to the cross bar switch 310 through respective local memory controller buses 326a, 326b. The cross bar switch 310, which may be of a conventional or hereinafter developed design, can couple any of the link interfaces 302, 304, 306, 308 to each other. The link interfaces 302, 304, 306, 308 may be either unidirectional or duplex interfaces, and the nature of the memory accesses coupled to or from the link interfaces 302, 304, 306, 308 may vary as desired, including communication protocols having conventional memory address, control and data signals, shared address and control signals and packetized memory access signals. As shown in Figure 3, the link interfaces 302 and 304 are coupled to the downstream bus 154 and the link interfaces 306 and 308 are coupled to the upstream bus 156.

[024] The cross bar switch 310 can also couple any of the link interfaces 302, 304, 306, 308 to either or both of the memory controllers 324a, 324b, each of which is coupled to a plurality of memory devices 148 (not shown in Figure 3) over respective local memory buses 150 (Figure 2). The memory controllers 324a, 324b may be conventional memory

controllers or some hereinafter developed design for a memory controller. The specific structure and operation of the memory controllers 324a, 324b will, of course, depend on the nature of the memory devices 148 used in the memory modules 130a-c. The cross bar switch 310 couples the link interfaces 302, 304, 306, 308 to the memory controllers 324a, 324b to allow any of a plurality of memory access devices to write data to or read data from the memory devices 148 coupled to the memory controllers 324a, 324b. The cross bar switch 310 further couples the link interfaces 302, 304, 306, 308 to the memory controllers 324a, 324b to allow any data to be transferred to or from the memory devices 148 coupled to the memory controllers 324a- 324b from or to, respectively, other memory modules 130a-c containing a memory hub 240. Thus, as previously discussed, the memory hub 240 is capable of receiving memory requests and providing memory responses in both downstream and upstream directions over the downstream and upstream buses 154, 156.

[025] It will be appreciated by those ordinarily skilled in the art that Figure 3 illustrates merely a portion of the memory hub 240, and that the memory hub 240 will generally include components in addition to those shown in Figure 3. For example, a cache memory for each of the memory controllers 324a, 324b can be included for storing recently or frequently accessed data retrieved from or stored in the memory devices 148. Additionally, a write buffer can also be included for accumulating write addresses and data directed to the memory devices 148 serviced by a respective one of the memory controllers 324a, 324b if the memory devices 148 are busy servicing a read memory request or other read requests are pending. Such components are conventional and known in the art. These components have been omitted from Figure 3 in the interest of brevity and clarity. It will further be appreciated by those ordinarily skilled in the art that in some applications, components shown in Figure 3 may be omitted. For example, although the memory hub 240 shown in Figure 3 includes two memory controllers 324a, 324b the number of memory controllers may vary as desired.

[026] Figure 4 illustrates a processor-based computing system 400 according to another embodiment of the present invention. The system 400 includes many of the same functional blocks as previously described with reference to Figures 1 and 2. As such, the

same reference numbers will be used in Figure 4 as in Figures 1 and 2 to refer to the same functional blocks where appropriate. The system 400 includes a processor 104 coupled to a memory hub controller 428 through a processor bus 106. A cache memory 108 is also coupled to the processor bus 106 to provide the processor 104 with temporary storage of frequently used data and instructions. The memory hub controller 428 is further coupled to a system controller 110, which serves as a communications path to the processor 104 for a variety of other components. As shown in Figure 4, data storage device 124 is coupled to the system controller 110 to allow the processor 104 to store data or retrieve data from internal or external storage media (not shown).

[027] The memory hub controller 428 is coupled over a high speed bi-directional or unidirectional system controller/hub interface 134 to several memory modules 130a-c. The controller/hub interface 134 includes a downstream bus 154 and an upstream bus 156 which are used to couple data, address, and/or control signals away from or toward, respectively, the memory hub controller 428. Each memory module 130a-c in the system 400 includes a memory hub 240 that is coupled to the system controller/hub interface 134, and which is further coupled a number of memory devices 148 through command, address and data buses, collectively shown as bus 150. The memory hub 240 efficiently routes memory requests and responses between the memory hub controller 128 and the memory devices 148. As with the memory hub 240 shown in Figure 2, memory requests and memory responses can be provided in both downstream and upstream directions over the downstream and upstream buses 154, 156, respectively, by the memory hub 240.

[028] Coupled in series with the memory modules 130a-c over the downstream and upstream buses 154, 156 are component expansion modules 230 and 430. The component expansion module 230, as previously described with reference to Figure 2, includes a graphics controller 234 coupled to local memory devices 248 over a local graphics/memory bus 250. The component expansion module 230 provides video data over a video bus 260 to a video terminal (not shown), as known in the art. In contrast to the system 200 of Figure 2, the system 400 further includes the component expansion module 430. The component expansion module 430 includes an input/output (IO) processor 434 coupled to

local memory devices 448 over a local memory device bus 450. Although the component expansion module 430 includes local memory devices 448, the IO processor 434 has access to system memory, for example, memory modules 130a-c, as well.

[029] Unlike the systems 100 and 200, where the input and output devices 118, 120 are coupled to the system controller 110, input and output devices (not shown in Figure 4) can be coupled to the system 400 through the component expansion module 430 and a high-speed IO bus 460. By including the component expansion module 430, memory request and response loading on the system controller 410 can be reduced compared to the configuration of systems 100 and 200. Using a consistent communication protocol with the memory hub 240 over the downstream and upstream buses 154, 156, the memory hub controller 428, the IO processor 434, and the graphics controller 234, can each access the memory modules 130a-c independently. As shown in Figure 4, the memory modules 130a-c and the component expansion modules 230, 430 are series coupled in an arrangement that takes advantage of the point-to-point architecture provided by the downstream and upstream buses 154, 156. The memory hub controller 428, the IO processor 434 and the graphics controller 234 each have a respective memory module 130a-c which can be used primarily for servicing memory requests by the respective component. That is, the memory module 130a can be used primarily by the memory hub controller 428 for servicing memory requests from the processor 104 and the system controller 410, the memory module 130b can be used primarily by the component expansion module 430 for servicing memory requests from the IO processor 434, and the memory module 130c can be used primarily by the component expansion module 230 for servicing memory requests from the graphics controller 234. Thus, although the memory hub controller 428, the component expansion module 430, and the component expansion module 230 have access to any of the memory modules 130a-c, memory requests from each of the requesting entities can be primarily serviced by a respective memory module 130a-c. As a result, the memory request and response loading that is conventionally handled by the system controller 110 is distributed throughout the memory system, thereby reducing the likelihood of memory requests and response being bottlenecked through one access point.

[030] It will be appreciated by those ordinarily skilled in the art that the embodiments shown in Figures 2 and 4 have been provided by way of example, and are not intended to limit the scope of the present invention. Modifications can be made to the previously described embodiments without departing from the scope of the present invention. For example, the system 400 has been described as providing each of the requesting components, the memory hub controller 428, the component expansion module 430, and the component expansion module 230, with a respective memory module 130a-c for primarily servicing memory requests. However, only portions of the memory available on a memory module 130a-c can be used for one requesting entity, with the remaining memory of the same memory module 130a-c allocated for primarily servicing the memory requests of another requesting entity. That is, the allocation of memory is not limited to a per module basis, but can be allocated as desired. Additionally, the order in which the memory modules 130a-c and the requesting entities are coupled, namely the memory hub controller 428, the component expansion module 430, and the component expansion module 230, can be changed and remain within the scope of the present invention. Although the order of the requesting entities can be arranged advantageously with respect to the memory modules 130a-c, as previously described with respect to having a primary memory for servicing memory requests, the present invention is not limited to any specific order of coupling of the memory modules and requesting entities.

[031] From the foregoing it will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without deviating from the spirit and scope of the invention. Accordingly, the invention is not limited except as by the appended claims.